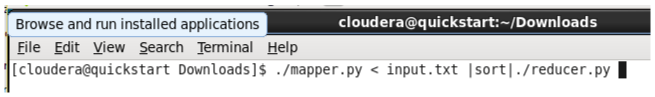
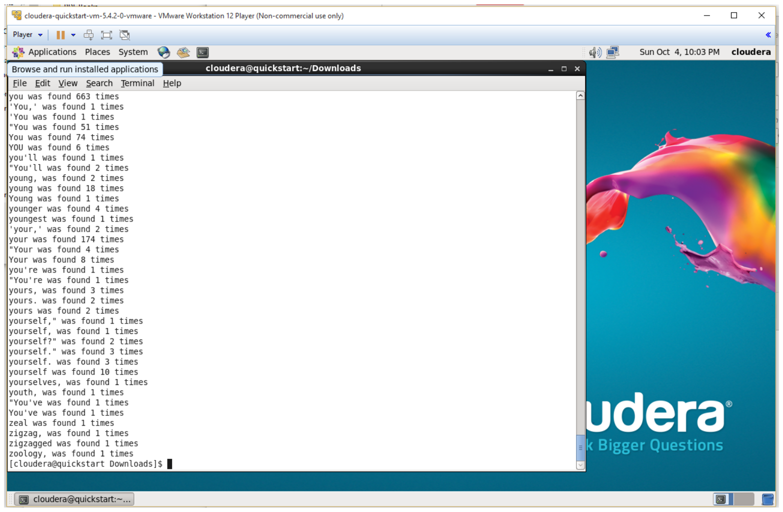
MSDS600 Week 7 Assignment - Nathan Worsham



```
[cloudera@quickstart Downloads]$ ./mapper.py < input.txt |sort|./reducer.py
```

I worked through the instructions for this assignment running the python code as is:

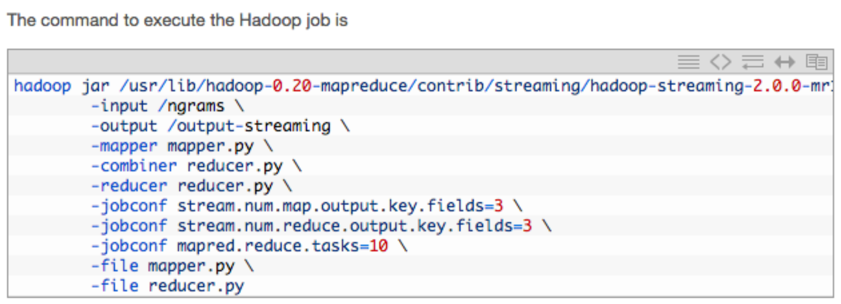


```
you was found 663 times
'You,' was found 1 times
'You was found 1 times
"You was found 51 times
You was found 74 times
YOU was found 6 times
you'll was found 1 times
"You'll was found 2 times
young, was found 2 times
young was found 18 times
Young was found 1 times
younger was found 4 times
youngest was found 1 times
'your,' was found 2 times
your was found 174 times
"Your was found 4 times
Your was found 8 times
you're was found 1 times
"You're was found 1 times
yours, was found 3 times
yours. was found 2 times
yours was found 2 times
yourself," was found 1 times
yourself, was found 1 times
yourself?" was found 2 times
yourself." was found 3 times
yourself. was found 3 times
yourself was found 10 times
yourselves, was found 1 times
youth, was found 1 times
"You've was found 1 times
You've was found 1 times
zeal was found 1 times
zigzag, was found 1 times
zigzagged was found 1 times
zoology, was found 1 times
[cloudera@quickstart Downloads]$
```

I ran into just a couple of minor issues. The first was getting the files to the VM machine. I ended up using the built-in Firefox browser to go to the worldclass.regis.edu site to get the files. The other issue I ran into is that the files were not executable, so for the files that needed to be executable—mapper and reduce—I ran the command:

```
chmod 755 filename
```

Next, I was having trouble understanding what the assignment wanted us to accomplish. I do not really know how to write any Java code so I started by Googling for a "Hadoop Python SDK". This brought me to a blog post on Cloudera's site called "A Guide to Python Frameworks for Hadoop" by Uri Laserson (2013). In this post it had a section on Hadoop Streaming, which from the reading I had learned that the streaming API allows non-Java programing languages to be used to interact with the Hadoop system. The example python scripts on the site were even named the same—mapper.py and reducer.py—so I felt this was the right route to go and where I spent my time trying to get to work. Later I learned that really just running the python scripts was the assignment, but I did not feel that gave me any understanding on how to use Hadoop as the supplied scripts were really just running some already written python code piped through a bash command and piped again to another python script. Since the Cloudera example uses a mapper and reducer script and I was already given similar python files, I mostly concentrated on the example they gave to execute the command:

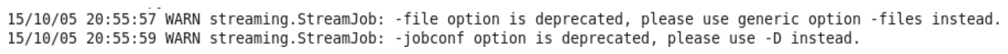

The command to execute the Hadoop job is

```
hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr
        -input /ngrams \
        -output /output-streaming \
        -mapper mapper.py \
        -combiner reducer.py \
        -reducer reducer.py \
        -jobconf stream.num.map.output.key.fields=3 \
        -jobconf stream.num.reduce.output.key.fields=3 \
        -jobconf mapred.reduce.tasks=10 \
        -file mapper.py \
        -file reducer.py
```

I realized that my reducer.py file would need to be edited to produce a different output as their reducer script example gave a tab separated output. I copied the output line, commented out the original and changed it so that it just outputted a key and value separated by a tab:

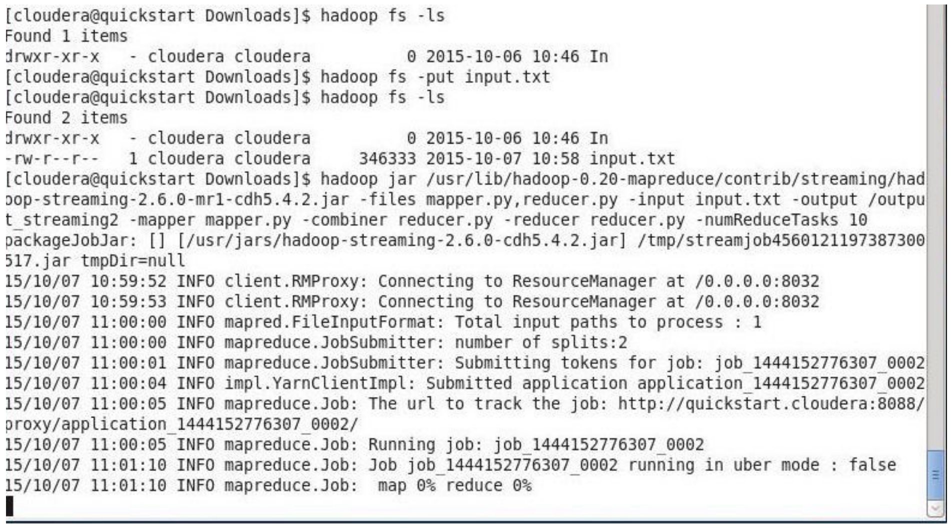
```
#print previous_key+" was found "+str(total)+" times"
print previous_key+"\t"+str(total)
```

Trying to run roughly the same command minus the key field settings because in their example the key had tabs in it. I quickly found that some of these options they showed were old or deprecated options from the help file that appeared on the screen because I ran the command incorrectly. Instead of having to specify each script on its own option (`-file`), the new option was `-files` and you give it a comma separated list, and the new option has to go at the beginning of the command. The help file mentioned that this option was so that each node gets the script and can execute it. Next I found that the `-jobconf` option was deprecated, the example was trying to use this to set the number of reduce tasks, I found the option `-numReduceTasks` instead.



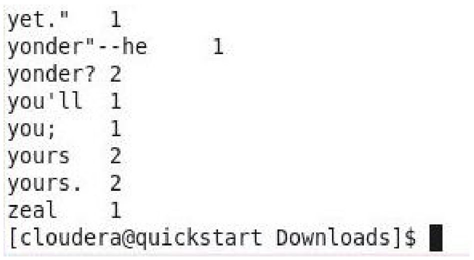
```
15/10/05 20:55:57 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
15/10/05 20:55:59 WARN streaming.StreamJob: -jobconf option is deprecated, please use -D instead.
```

Now that I had my command correct I was still getting errors that it could not find my inputs.txt file as it was giving me an error about "input path does not exist". I was very confused here as I wasn't sure how to get it my file (I tried giving it a fully qualified path and placing the file in the home directory of the user to no avail). Stackoverflow.com to the rescue! In the post I found, I realized that you treat the Hadoop file system almost as if it were a web or ftp site, you have to upload your input file to it using the `hadoop fs` command. So I then used `hadoop fs -put input.txt` to upload my input file. Now my command finally started to work:



```
[cloudera@quickstart Downloads]$ hadoop fs -ls
Found 1 items
drwxr-xr-x   - cloudera cloudera          0 2015-10-06 10:46 In
[cloudera@quickstart Downloads]$ hadoop fs -put input.txt
[cloudera@quickstart Downloads]$ hadoop fs -ls
Found 2 items
drwxr-xr-x   - cloudera cloudera          0 2015-10-06 10:46 In
-rw-r--r--   1 cloudera cloudera     346333 2015-10-07 10:58 input.txt
[cloudera@quickstart Downloads]$ hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/had
oop-streaming-2.6.0-mr1-cdh5.4.2.jar -files mapper.py,reducer.py -input input.txt -output /outpu
t_streaming2 -mapper mapper.py -combiner reducer.py -reducer reducer.py -numReduceTasks 10
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.4.2.jar] /tmp/streamjob4560121197387300
517.jar tmpDir=null
15/10/07 10:59:52 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
15/10/07 10:59:53 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
15/10/07 11:00:00 INFO mapred.FileInputFormat: Total input paths to process : 1
15/10/07 11:00:00 INFO mapreduce.JobSubmitter: number of splits:2
15/10/07 11:00:01 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1444152776307_0002
15/10/07 11:00:04 INFO impl.YarnClientImpl: Submitted application application_1444152776307_0002
15/10/07 11:00:05 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/
proxy/application_1444152776307_0002/
15/10/07 11:00:05 INFO mapreduce.Job: Running job: job_1444152776307_0002
15/10/07 11:01:10 INFO mapreduce.Job: Job job_1444152776307_0002 running in uber mode : false
15/10/07 11:01:10 INFO mapreduce.Job:  map 0% reduce 0%
```

I suppose I was surprised how long it took to run the MapReduce job. I know that this is running on a VM which would cause performance issues but in "real life" I work with a product called Splunk which also uses the concept of MapReduce, and on Splunk no matter if you are on a single node or many the operation to ingest (and retrieve) data is nearly realtime. Last I wanted a way to check that my data was truly in the HDFS. I was able to find that the same `fs` command allows you to "cat" (which actually stands for concatenate) a file or group of files. So using `hadoop fs -cat /output_streaming/*` I was able to see my results which looked very similar to the output of the python script example:



```
yet."   1
yonder"--he     1
yonder? 2
you'll  1
you;    1
yours   2
yours.  2
zeal    1
[cloudera@quickstart Downloads]$
```

References

Laserson, Uri, 2013. A Guide to Python Frameworks for Hadoop. Retrieved from http://blog.cloudera.com/blog/2013/01/a-guide-to-python-frameworks-for-hadoop/
Stackoverlow.com, 2013. Retrieved from http://stackoverflow.com/questions/15191832/first-hadoop-project-error-input-path-does-not-exist